

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24
- 25

2
3

4

5
6

7

8
911
12
13

14

15
16
17
18
19

20
21
22

24

1 previous median, wherein a median of a first right side grouping is linked to the
2 parent node;

3 wherein when a grouping has an even number of elements, the median is a
4 left element of two middle values of the grouping;

5 wherein when a grouping has an odd number of number of elements, the
6 median is a middle value element of the grouping; and

7 wherein the elements of the lists include logged events.

8 Claim 2 (original): A computer-readable medium having stored thereon
9 computer-executable instructions for performing the method of claim 1.

10 Claim 3 (previously amended): The method of claim 1, wherein each
11 element in the list includes a pointer to a corresponding node of a plurality of
12 nodes in a partially assembled binary tree, wherein each node has a left child
13 pointer, and wherein creating the left side nodes further comprises assigning a
14 value to the left child pointer of at least one of the nodes.

15 Claim 4 (previously amended): The method of claim 1, wherein each
16 element in the list includes a pointer to a corresponding node of a plurality of
17 nodes in a partially assembled binary tree, wherein each node has a right child
18 pointer, and wherein creating the right side nodes further comprises assigning a
19 value to the right child pointer of at least one of the nodes.

20 Claim 5 (currently amended): The method of claim 1, wherein creating the
21 left side descendent nodes comprises ~~inserting-linking~~ the left side descendent
22 nodes ~~into-to~~ a partially assembled version of the binary tree, wherein creating the
23 right side descendent nodes comprises ~~creating-linking~~ the right side descendent
24 nodes ~~into-to~~ the partially assembled version of the binary tree, and wherein the
25 list is a linked list that acts as a wrapper around the partially assembled version of

1 the binary tree.

2 Claim 6 (canceled).

3
4 Claim 7 (currently amended): A method for creating a binary tree data
5 structure, the data structure embodied in a computer-readable medium, from an
6 ordered list of at least four elements, each element having an associated value in
7 the list, comprising:

8 determining whether the list has an even or odd number of elements;

9 separating the list into left side groupings and right side groupings based
10 on whether the list has an even or odd number of elements, the groupings being
11 groupings separated by a parent node defined by a median of the list, wherein the
12 median is a left element of two middle values of the list when the list has an even
13 number of elements, or the median is a middle value element of the list when list
14 has an odd number of elements;

15 creating left side descendent nodes of the binary tree by successively
16 finding a median of each left side grouping and linking each found median to the
17 previous median, wherein a median of a first left side grouping is linked to the
18 parent node;

19 creating right side descendent nodes of the binary tree by successively
20 finding a median of each right side grouping and linking each found median to the
21 previous median, wherein a median of a first right side grouping is linked to the
22 parent node;

23 wherein when a grouping has an even number of elements, the median is a
24 left element of two middle values of the grouping;

25 wherein when a grouping has an odd number of number of elements, the
median is a middle value element of the grouping; and

wherein the elements of the list include data representing number of times
one or more threads of execution have passed through one or more code modules.

1 Claim 8 (currently amended): A method for creating a binary tree data
2 structure, the data structure embodied in a computer-readable medium, from an
3 ordered list of at least four elements, each element having an associated value in
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;

6 separating the list into left side groupings and right side groupings based on
7 whether the list has an even or odd number of elements, the groupings being
8 separated by a parent node defined by a median of the list, wherein the median is a
9 left element of two middle values of the list when the list has an even number of
10 elements, or the median is a middle value element of the list when the list has an
11 odd number of elements;

12 creating left side descendent nodes of the binary tree by successively
13 finding a median of each left side grouping and linking each found median to the
14 previous median, wherein a median of a first left side grouping is linked to the
15 parent node;

16 creating right side descendent nodes of the binary tree by successively
17 finding a median of each right side grouping and linking each found median to the
18 previous median, wherein a median of a first right side grouping is linked to the
19 parent node;

20 wherein when a grouping has an even number of elements, the median is a
21 left element of two middle values of the grouping;

22 wherein when a grouping has an odd number of number of elements, the
23 median is a middle value element of the grouping; and

24 wherein the created right and left descendant nodes include data
25 representing a number of times one or more threads of execution have passed
through one or more code modules.

1 Claim 9 (currently amended): A method for creating a binary tree data
2 structure, the data structure embodied in a computer-readable medium, from an
3 ordered list of at least four elements, each element having an associated value in
4 the list, comprising:

5 separating the list into left side groupings and right side groupings based on
6 whether the list has an even or odd number of elements, the groupings being
7 separated by a parent node defined by a median of the list, wherein the median is a
8 left element of two middle values of the list when the list has an even number of
9 elements, or the median is a middle value element of the list when the list has an
10 odd number of elements;

11 creating left side descendent nodes of the binary tree by successively
12 finding a median of each left side grouping and linking each found median to the
13 previous median, wherein a median of a first left side grouping is linked to the
14 parent node;

15 creating right side descendent nodes of the binary tree by successively
16 finding a median of each right side grouping and linking each found median to the
17 previous median, wherein a median of a first right side grouping is linked to the
18 parent node;

19 wherein when a grouping has an even number of elements, the median is a
20 left element of two middle values of the grouping;

21 wherein when a grouping has an odd number of number of elements, the
22 median is a middle value element of the grouping; and

23 wherein the created right and left descendant nodes include one or more
24 pointers to data representing a number of times one or more threads of execution
25 have passed through one or more code modules.

26 Claim 10 (original): The method of claim 1, wherein the list is an ordered
27 linked list.

1 Claim 11 (canceled).

2 Claim 12 (canceled).

3
4 Claim 13 (canceled).

5
6 Claim 14 (canceled).

7
8 Claim 15 (currently amended): A method for creating a binary tree data
9 structure, the data structure embodied in a computer-readable medium, from an
10 ordered list of at least four elements, each element having an associated value in
11 the list, comprising:

12 determining whether the list has an even or odd number of elements;

13 separating the list into left side groupings and right side groupings based on
14 whether the list has an even or odd number of elements, the groupings being
15 separated by a parent node defined by a median of the list, wherein the median is a
16 left element of two middle values of the list when the list has an even number of
17 elements, or the median is a middle value element of the list when the list has an
18 odd number of elements;

19 creating right side descendent nodes of the binary tree by successively
20 finding a median of each right side grouping and linking each found median to the
21 previous median, wherein a median of a first left side grouping is linked to the
22 parent node;

23 creating left side descendent nodes of the binary tree by successively
24 finding a median of each left side grouping and linking each found median to the
25 previous median, wherein a median of a first left side grouping is linked to the
parent node;

wherein when a grouping has an even number of elements, the median is a
right element of two middle values of the grouping;

1 wherein when a grouping has an odd number of number of elements, the
2 median is a middle value element of the grouping; and

3 wherein the elements of the list include logged events.

4 Claim 16 (original): A computer-readable medium having stored thereon
5 computer-executable instructions for performing the method of claim 15.

6
7 Claim 17 (previously amended): The method of claim 15, wherein each
8 element in the list includes a pointer to a corresponding node of a plurality of
9 nodes in a partially assembled binary tree, wherein each node has a right child
10 pointer, and wherein creating the right side nodes further comprises assigning a
11 value to the right child pointer of at least one of the nodes.

12 Claim 18 (previously amended): The method of claim 15, wherein each
13 element in the list includes a pointer to a corresponding node of a plurality of
14 nodes in a partially assembled binary tree, wherein each node has a left child
15 pointer, and wherein creating the left side nodes further comprises assigning a
16 value to the left child pointer of at least one of the nodes.

17 Claim 19 (currently amended): The method of claim 15, wherein creating
18 the right side descendent nodes comprises ~~creating-linking~~ the right side
19 descendent nodes ~~into-to~~ a partially assembled version of the binary tree, wherein
20 creating the left side descendent nodes comprises ~~creating-linking~~ the left side
21 descendent nodes ~~into-to~~ the partially assembled version of the binary tree, and
22 wherein the list is a linked list that acts as a wrapper around the partially
23 assembled version of the binary tree.

24 Claim 20 (canceled).
25

1 Claim 21 (currently amended): A method for creating a binary tree data
2 structure, the data structure embodied in a computer-readable medium, from an
3 ordered list of at least four elements, each element having an associated value in
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;

6 separating the list into left side groupings and right side groupings based on
7 whether the list has an even or odd number of elements, the groupings being
8 separated by a parent node defined by a median of the list, wherein the median is a
9 left element of two middle values of the list when the list has an even number of
10 elements, or the median is a middle value element of the list when the list has an
11 odd number of elements;

12 creating right side descendent nodes of the binary tree by successively
13 finding a median of each right side grouping and linking each found median to the
14 previous median, wherein a median of a first left side grouping is linked to the
15 parent node;

16 creating left side descendent nodes of the binary tree by successively
17 finding a median of each left side grouping and linking each found median to the
18 previous median, wherein a median of a first right side grouping is linked to the
19 parent node;

20 wherein when a grouping has an even number of elements, the median is a
21 right element of two middle values of the grouping;

22 wherein when a grouping has an odd number of number of elements, the
23 median is a middle value element of the grouping; and

24 wherein the elements of the list include data representing a number of times
25 one or more threads of execution have passed through one or more code modules.

1 Claim 22 (currently amended): A method for creating a binary tree data
2 structure, the data structure embodied in a computer-readable medium, from an
3 ordered list of at least four elements, each element having an associated value in
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;

6 separating the list into left side groupings and right side groupings based on
7 whether the list has an even or odd number of elements, the groupings being
8 separated by a parent node defined by a median of the list, wherein the median is a
9 left element of two middle values of the list when the list has an even number of
10 elements, or the median is a middle value element of the list when the list has an
11 odd number of elements;

12 creating right side descendent nodes of the binary tree by successively
13 finding a median of each right side grouping and linking each found median to the
14 previous median, wherein a median of a first left side grouping is linked to the
15 parent node;

16 creating left side descendent nodes of the binary tree by successively
17 finding a median of each left side grouping and linking each found median to the
18 previous median, wherein a median of a first right side grouping is linked to the
19 parent node;

20 wherein when a grouping has an even number of elements, the median is a
21 right element of two middle values of the grouping;

22 wherein when a grouping has an odd number of number of elements, the
23 median is a middle value element of the grouping; and

24 wherein the created right and left descendant nodes include data
25 representing a number of times one or more threads of execution have passed
through one or more code modules.

1 Claim 23 (currently amended): A method for creating a binary tree data
2 structure, the data structure embodied in a computer-readable medium, from an
3 ordered list of at least four elements, each element having an associated value in
4 the list, comprising:

5 determining whether the list has an even or odd number of elements;

6 separating the list into left side groupings and right side groupings based on
7 whether the list has an even or odd number of elements, the groupings being
8 separated by a parent node defined by a median of the list, wherein the median is a
9 left element of two middle values of the list when the list has an even number of
10 elements, or the median is a middle value element of the list when the list has an
11 odd number of elements;

12 creating right side descendent nodes of the binary tree by successively
13 finding a median of each right side grouping and linking each found median to the
14 previous median, wherein a median of a first left side grouping is linked to the
15 parent node;

16 creating left side descendent nodes of the binary tree by successively
17 finding a median of each left side grouping and linking each found median to the
18 previous median, wherein a median of a first right side grouping is linked to the
19 parent node;

20 wherein when a grouping has an even number of elements, the median is a
21 right element of two middle values of the grouping;

22 wherein when a grouping has an odd number of number of elements, the
23 median is a middle value element of the grouping; and

24 wherein the created right and left descendant nodes include one or more
25 pointers to data representing a number of times one or more threads of execution
have passed through one or more code modules.

Claim 24 (original): The method of claim 15, wherein the list is an ordered
linked list.

1 Claim 25 (canceled).

2
3 Claim 26 (canceled).

4
5 Claim 27 (canceled).

6
7 Claim 28 (canceled).

8 Claim 29 (previously amended): The method of claim 7, wherein each
9 element in the list includes a pointer to a corresponding node of a plurality of
10 nodes in a partially assembled binary tree, wherein each node has a left child
11 pointer, and wherein creating the left side nodes further comprises assigning a
12 value to the left child pointer of at least one of the nodes.

13 Claim 30 (previously amended): The method of claim 7 wherein each
14 element in the list includes a pointer to a corresponding node of a plurality of
15 nodes in a partially assembled binary tree, wherein each node has a right child
16 pointer, and wherein creating the right side nodes further comprises assigning a
17 value to the right child pointer of at least one of the nodes.

18 Claim 31 (currently amended): The method of claim 7, wherein creating
19 the left side descendent nodes comprises ~~creating-linking~~ the left side descendent
20 nodes ~~into-to~~ a partially assembled version of the binary tree, wherein creating the
21 right side descendent nodes comprises ~~creating-linking~~ the right side descendent
22 nodes ~~into-to~~ the partially assembled version of the binary tree, and wherein the
23 list is a linked list that acts as a wrapper around the partially assembled version of
24 the binary tree.
25

1 Claim 32 (previously presented): The method of claim 7, wherein the list is
2 an ordered linked list.

3 Claim 33 (previously presented): A computer-readable medium having
4 stored thereon computer-executable instructions for performing the method of
5 claim 7.

6 Claim 34 (previously amended): The method of claim 8, wherein each
7 element in the list includes a pointer to a corresponding node of a plurality of
8 nodes in a partially assembled binary tree, wherein each node has a left child
9 pointer, and wherein creating the left side nodes further comprises assigning a
10 value to the left child pointer of at least one of the nodes.

11 Claim 35 (previously amended): The method of claim 8, wherein each
12 element in the list includes a pointer to a corresponding node of a plurality of
13 nodes in a partially assembled binary tree, wherein each node has a right child
14 pointer, and wherein creating the right side nodes further comprises assigning a
15 value to the right child pointer of at least one of the nodes.

16 Claim 36 (currently amended): The method of claim 8, wherein creating
17 left side descendent nodes comprises ~~inserting-linking~~ the left side descendent
18 nodes ~~into-to~~ a partially assembled version of the binary tree, wherein creating the
19 right side descendent nodes comprises ~~creating-linking~~ the right side descendent
20 nodes ~~into-to~~ the partially assembled version of the binary tree, and wherein the
21 list is a linked list that acts as a wrapper around the partially assembled version of
22 the binary tree.

23 Claim 37 (previously presented): The method of claim 8, wherein the list is
24 an ordered linked list.
25

1 Claim 38 (previously presented): A computer-readable medium having
2 stored thereon computer-executable instructions for performing the method of
3 claim 8.

4
5 Claim 39 (previously amended): The method of claim 9, wherein each
6 element in the list includes a pointer to a corresponding node of a plurality of
7 nodes in a partially assembled binary tree, wherein each node has a left child
8 pointer, and wherein creating the left side nodes further comprises assigning a
9 value to the left child pointer of at least one of the nodes.

10 Claim 40 (previously amended): The method of claim 9, wherein each
11 element in the list includes a pointer to a corresponding node of a plurality of
12 nodes in a partially assembled binary tree, wherein each node has a right child
13 pointer, and wherein creating the right side nodes further comprises assigning a
14 value to the right child pointer of at least one of the nodes.

15 Claim 41 (currently amended): The method of claim 9, wherein creating
16 the left side descendent nodes comprises ~~creating-linking~~ the left side descendent
17 nodes ~~into-to~~ a partially assembled version of the binary tree, wherein creating the
18 right side descendent nodes comprises ~~creating-linking~~ the right side descendent
19 nodes ~~into-to~~ the partially assembled version of the binary tree, and wherein the
20 list is a linked list that acts as a wrapper around the partially assembled version of
21 the binary tree.

22 Claim 42 (previously presented): The method of claim 9, wherein the list is
23 an ordered linked list.
24
25

1 Claim 43 (previously presented): A computer-readable medium having
2 stored thereon computer-executable instructions for performing the method of
3 claim 9.

4 Claim 44 (canceled).

5
6 Claim 45 (canceled).

7
8 Claim 46 (canceled).

9
10 Claim 47 (canceled).

11 Claim 48 (previously amended): The method of claim 21, wherein each
12 element in the list includes a pointer to a corresponding node of a plurality of
13 nodes in a partially assembled binary tree, wherein each node has a right child
14 pointer, and wherein creating the right side nodes further comprises assigning a
15 value to the right child pointer of at least one of the nodes..

16 Claim 49 (previously amended): The method of claim 21, wherein each
17 element in the list includes a pointer to a corresponding node of a plurality of
18 nodes in a partially assembled binary tree, wherein each node has a left child
19 pointer, and wherein creating the left side nodes further comprises assigning a
20 value to the left child pointer of at least one of the nodes.

21 Claim 50 (currently amended): The method of claim 21, wherein creating
22 the right side descendent nodes comprises ~~creating-linking~~ the right side
23 descendent nodes ~~into-to~~ a partially assembled version of the binary tree, wherein
24 creating the left side descendent nodes comprises ~~creating-linking~~ the left side
25 descendent nodes ~~into-to~~ the partially assembled version of the binary tree, and

1 wherein the list is a linked list that acts as a wrapper around the partially
2 assembled version of the binary tree.

3 Claim 51 (previously presented): The method of claim 21, wherein the list
4 is an ordered linked list.

5
6 Claim 52 (previously presented): A computer-readable medium having
7 stored thereon computer-executable instructions for performing the method of
8 claim 21.

9 Claim 53 (previously amended): The method of claim 22, wherein each
10 element in the list includes a pointer to a corresponding node of a plurality of
11 nodes in a partially assembled binary tree, wherein each node has a right child
12 pointer, and wherein creating the right side nodes further comprises assigning a
13 value to the right child pointer of at least one of the nodes.

14 Claim 54 (previously amended): The method of claim 22, wherein each
15 element in the list includes a pointer to a corresponding node of a plurality of
16 nodes in a partially assembled binary tree, wherein each node has a left child
17 pointer, and wherein creating the left side nodes further comprises assigning a
18 value to the left child pointer of at least one of the nodes.

19 Claim 55 (currently amended): The method of claim 22, wherein creating
20 the right side descendent nodes comprises ~~creating-linking~~ the right side
21 descendent nodes ~~into-to~~ a partially assembled version of the binary tree, wherein
22 creating the left side descendent nodes comprises ~~creating-linking~~ the left side
23 descendent nodes ~~into-to~~ the partially assembled version of the binary tree, and
24 wherein the list is a linked list that acts as a wrapper around the partially
25 assembled version of the binary tree.

1 Claim 56 (previously presented): The method of claim 22, wherein the list
2 is an ordered linked list.
3

4 Claim 57 (previously presented): The computer-readable medium having
5 stored thereon computer-executable instructions for performing the method of
6 claim 22.
7

8 Claim 58 (previously amended): The method of claim 23, wherein each
9 element in the list includes a pointer to a corresponding node of a plurality of
10 nodes in a partially assembled binary tree, wherein each node has a right child
11 pointer, and wherein creating the right side nodes further comprises assigning a
12 value to the right child pointer of at least one of the nodes.

13 Claim 59 (previously amended): The method of claim 23, wherein each
14 element in the list includes a pointer to a corresponding node of a plurality of
15 nodes in a partially assembled binary tree, wherein each node has a left child
16 pointer, and wherein creating the left side nodes further comprises assigning a
17 value to the left child pointer of at least one of the nodes.

18 Claim 60 (currently amended): The method of claim 23, wherein creating
19 the right side descendent nodes comprises ~~creating-linking~~ the right side
20 descendent nodes ~~into-to~~ a partially assembled version of the binary tree, wherein
21 creating the left side descendent nodes comprises ~~creating-linking~~ the left side
22 descendent nodes ~~into-to~~ the partially assembled version of the binary tree, and
23 wherein the list is a linked list that acts as a wrapper around the partially
24 assembled version of the binary tree.
25

1 Claim 61 (previously presented): The method of claim 23, wherein the list
2 is an ordered linked list.

3 Claim 62 (previously presented): A computer-readable medium having
4 stored thereon computer-executable instructions for performing the method of
5 claim 23.

6
7 Claim 63 (currently amended): A method for creating a binary tree data
8 structure, the data structure embodied in a computer-readable medium, from an
9 ordered list of at least four elements, each element having an associated value in
10 the list, comprising:

11 (a) determining whether the list has an even or odd number of elements;

12 (b) designating a median element of the list as a parent element based on
13 whether the list has an even or odd number of elements, wherein the parent
14 element divides the list into left side groupings and right side groupings, wherein
15 the median is a right element of two middle values of the list when the list has an
16 even number of elements, or the median is a middle value element of the list when
17 the list has an odd number of elements;

18 (c) successively subdividing the right side groupings of the list and linking
19 each successive median element with a previous median element, thereby creating
20 right side descendent nodes in the binary tree, and wherein a median of a first right
21 side grouping is linked to the parent element;

22 (d) once each right side grouping has been exhausted as a result of step (c),
23 stepping back up the tree through each successive ancestor node until reaching an
24 element having left side groupings in the list, and, upon reaching an element
25 having a left side grouping in the list, proceeding to step (e);

(e) subdividing the left side groupings and linking a median element of ~~the~~
a left side grouping with the element reached in step (d), thereby creating a left
side descendent of the binary tree;

1 (f) if the left side descendent of step (e) has a right side grouping in the list,
2 repeating step (c) for the right side grouping;

3 (g) if the left side descendent of step (e) has no right side groupings, but has
4 a left side grouping, repeating step (e) for the left side grouping;

5 wherein the median element of a grouping is a right element of two middle
6 values of the grouping when the grouping has an even number of elements, or the
7 median is a middle value element of the grouping when the list has an odd number
8 of number of elements; and

9 wherein the elements of the list include data representing number of times
10 one or more threads of execution have passed through one or more code modules.

11 Claim 64 (previously presented): A computer-readable medium having
12 stored thereon computer-executable instructions for performing the method of
13 claim 63.

14 Claim 65 (previously amended): A method for creating a binary tree data
15 structure, the data structure embodied in a computer-readable medium, from an
16 ordered list of at least four elements, each element having an associated value in
17 the list, comprising:

18 (a) determining whether the list has an even or odd number of elements;

19 (b) designating a median element of the list as a parent element based on
20 whether the list has an even or odd number of elements, wherein the parent
21 element divides the list into left side groupings and right side groupings, wherein
22 the median is a right element of two middle values of the list when the list has an
23 even number of elements, or the median is a middle value element of the list when
24 the list has an odd number of elements;

25 (c) determining if there are elements to the right of the parent element;

(d) if there are no elements to the right of the parent element, proceeding to
step (h);

1 (e) for the elements that are to the right of the parent element, finding a
median element;

2 (f) linking the median element of step (e) to the parent element so that the
3 median element is a child of the parent element;

4 (g) repeating steps (d) and (e), wherein the child element of step (f) is now
5 treated as the parent element in steps (d) and (e);

6 (h) locating a next element up on the tree that has elements to the left of it
and treating the element as a parent element in step (i);

7 (i) finding a median element of the elements to the left of the parent
8 element from step (h);

9 (j) linking the median element of step (i) to the parent element of step (h),
10 wherein the median element is a child of the parent;

11 (k) repeating steps (d) through (g), wherein the child element of step (j) is
12 treated as the parent element in step (d);

13 wherein the median element of a grouping is a right element of two middle
14 values of the grouping when the grouping has an even number of elements, or the
15 median is a middle value element of the grouping when the list has an odd number
of number of elements; and

16 wherein the elements of the list include data representing number of times
17 one or more threads of execution have passed through one or more code modules.

18
19 Claim 66 (previously presented): A computer-readable medium having
20 stored thereon computer-executable instructions for performing the method of
21 claim 65.
22
23
24
25